M 11

```
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
```

```
                                                                 M  4
**FILE**ID**SYMBOL












            SSSSSSSS  YY      YY  MM      MM  BBBBBBBB      OOOOOO    LL
            SSSSSSSS  YY      YY  MM      MM  BBBBBBBB      OOOOOO    LL
          SS          YY      YY  MMMM  MMMM  BB      BB  OO      OO  LL
          SS          YY      YY  MMMM  MMMM  BB      BB  OO      OO  LL
          SS            YY  YY    MM  MM  MM  BB      BB  OO      OO  LL
          SS            YY  YY    MM  MM  MM  BB      BB  OO      OO  LL
            SSSSSS        YY      MM      MM  BBBBBBBB    OO      OO  LL
            SSSSSS        YY      MM      MM  BBBBBBBB    OO      OO  LL
                  SS      YY      MM      MM  BB      BB  OO      OO  LL
                  SS      YY      MM      MM  BB      BB  OO      OO  LL
                  SS      YY      MM      MM  BB      BB  OO      OO  LL              ....
                  SS      YY      MM      MM  BB      BB  OO      OO  LL              ....
          SSSSSSSS        YY      MM      MM  BBBBBBBB      OOOOOO    LLLLLLLLLL      ....
          SSSSSSSS        YY      MM      MM  BBBBBBBB      OOOOOO    LLLLLLLLLL      ....


          LL            IIIIII      SSSSSSSS
          LL            IIIIII      SSSSSSSS
          LL              II      SS
          LL              II      SS
          LL              II      SS
          LL              II      SS
          LL              II        SSSSSS
          LL              II        SSSSSS
          LL              II              SS
          LL              II              SS
          LL              II              SS
          LL              II              SS
          LLLLLLLLLL    IIIIII    SSSSSSSS
          LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
0000            .TITLE  SYMBOL - SYMBOL TABLE MANIPULATION ROUTINES
0000            .IDENT  'V04-000'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000    ; SYMBOL TABLE MANIPULATION ROUTINES
0000    ;
0000    ; D. N. CUTLER 29-APR-77
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-006 HWS0031         Harold Schultz  14-Mar-1984
0000    ;               Add DELETE/SYMBOL/LOG
0000    ;
0000    ;       V03-005 PCG0009         Peter George    16-Aug-1983
0000    ;               Fix bug in binary symbol restoration logic.
0000    ;
0000    ;       V03-004 PCG0008         Peter George    27-May-1983
0000    ;               Add PTRDEF reference.
0000    ;
0000    ;       V03-003 PCG0007         Peter George    27-May-1983
0000    ;               Add DCL$DELSYM.
0000    ;
0000    ;       V03-002 PCG0006         Peter George    09-Mar-1983
0000    ;               Call DCL$FIND_KEYPAD.
0000    ;               Add DCL$RESTORE_SYM.
0000    ;
0000    ;       V03-001 PCG0005         Peter George    15-Nov-1982
0000    ;               Do roundup in DEADYNMEM.  Signal SPR error.
0000    ;---
```

```
                          0000  ;
                          0000  ; MACRO LIBRARY CALLS
                          0000  ;
                          0000          PRCDEF                          ;DEFINE PROCESS WORK AREA
                          0000          WRKDEF                          ;DEFINE COMMAND WORK AREA
                          0000          PTRDEF                          ;DEFINE TOKEN DESCRIPTORS
                          0000          SYMDEF                          ;DEFINE SYMBOL ENTRY OFFSETS
                          0000          IDFDEF                          ;DEFINE INDIRECT STACK OFFSETS
                          0000          CTXDEF                          ;DEFINE SPAWN CTX SYMBOLS
                          0000          $CLIMSGDEF                      ;DEFINE ERROR/STATUS VALUES
                          0000
                      00000000          .PSECT  DCL$ZCODE,BYTE,RD,NOWRT
                          0000  ;
                          0000  ;       SYMBOL TYPE DESCRIPTION STRINGS FOR DEL/SYM/LOG
                          0000  ;
       6C 61 63 6F 4C 00' 0000  LOCTAB: .ASCIC  /Local/
                      05  0000
                          0006
    6C 61 62 6F 6C 47 00' 0006  GBLTAB: .ASCIC  /Global/
                      06  0006
```

```
                   000D         .SBTTL  ALLOCATE AND INSERT ENTRY IN SYMBOL TABLE
                   000D   ;+
                   000D   ; DCL$ALLOCSYMABR - ALLOCATE AND INSERT ENTRY IN SYMBOL TABLE W/ ABBREVIATION
                   000D   ; DCL$ALLOCSYM - ALLOCATE AND INSERT ENTRY IN SYMBOL TABLE
                   000D   ;
                   000D   ; THIS ROUTINE IS CALLED TO ALLOCATE AND INSERT A SYMBOL ENTRY IN EITHER THE
                   000D   ; LOCAL OR GLOBAL SYMBOL TABLE.
                   000D   ;
                   000D   ; INPUTS:
                   000D   ;
                   000D   ;       R11 = ADDRESS OF PROCESS WORK AREA
                   000D   ;
                   000D   ;       R0 = TYPE OF SYMBOL VALUE (SYM_K_STRING OR SYM_K_BINARY)
                   000D   ;       R1/R2 = DESCRIPTOR OF SYMBOL VALUE
                   000D   ;       R3/R4 = DESCRIPTOR OF SYMBOL NAME.
                   000D   ;       R5 = ADDRESS OF SYMBOL TABLE LISTHEAD.
                   000D   ;
                   000D   ;       IT IS ASSUMED THAT CONTROL Y/C AST'S ARE DISABLED.
                   000D   ;
                   000D   ; OUTPUTS:
                   000D   ;
                   000D   ;       THE SPECIFIED SYMBOL TABLE IS SEARCHED FOR THE SPECIFIED ENTRY, AND
                   000D   ;       IF FOUND, THE OLD ENTRY IS DEALLOCATED. A SYMBOL TABLE ENTRY IS THEN
                   000D   ;       ALLOCATED, FILLED WITH THE SYMBOL AND VALUE INFORMATION, AND THEN
                   000D   ;       INSERTED IN THE SPECIFIED SYMBOL TABLE.
                   000D   ;
                   000D   ;       R0 LOW BIT CLEAR INDICATES ALLOCATION FAILURE WITH:
                   000D   ;
                   000D   ;               R0 = DCL$_SYMOVF - NO ROOM FOR SYMBOL DEFINITIONS.
                   000D   ;               R0 = DCL$_SYMDEL - ABBREVIATED SYMBOL NOT ALLOWED.
                   000D   ;               R0 = DCL$_ABSYMD - AMBIGUOUS SYMBOL DEFINITION.
                   000D   ;
                   000D   ;       R0 LOW BIT SET INDICATES SUCCESSFUL COMPLETION.
                   000D   ;
                   000D   ;               R1 = ADDRESS OF ALLOCATED SYMBOL ENTRY.
                   000D   ;               R2,R3,R4,R5 ARE DESTROYED.
                   000D   ;-
                   000D
                   000D   DCL$GT_SYMABR::                                 ;ABBREVIATED SYMBOL COUNTED STRING
            2A 00' 000D           .ASCIC  '*'
               01  000D
                   000F
                   000F   DCL$ALLOCSYMABR::                               ;ALLOCATE AND INSERT SYMBOL IN TABLE
         3F    BB  000F           PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ;SAVE SYMBOL ENTRY PARAMETERS
   64  53  2A  3A  0011           LOCC    #^A'*',R3,(R4)          ;FIND THE ABBREVIATION FLAG
         50    DD  0015           PUSHL   R0                      ;SAVE NUMBER OF EXTRA CHARACTERS
         22    13  0017           BEQL    ALLOCSYM                ;BRANCH IF NOT ABBREVIATED
         6E    D7  0019           DECL    (SP)                    ;REMOVE THE ABBR CHAR FROM EXTRA COUNT
      10 AE    D7  001B           DECL    16(SP)                  ;REMOVE THE ABBR CHAR FROM NAME COUNT
61  01 A1 6E   28  001E           MOVC    (SP),1(R1),(R1)         ;SLIDE REMAINING CHARACTERS OVER IT
         16    11  0023           BRB     ALLOCSYM
                   0025
                   0025   DCL$ALLOCSYM::                                  ;ALLOCATE AND INSERT SYMBOL IN TABLE
         3F    BB  0025           PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ;SAVE SYMBOL ENTRY PARAMETERS
         7E    D4  0027           CLRL    -(SP)                   ;SET NO ABBREVIATION
   64  53  2A  3A  0029           LOCC    #^A'*',R3,(R4)          ;FIND THE ABBREVIATION FLAG
         0C    13  002D           BEQL    ALLOCSYM                ;OK IF NONE FOUND
```

```
              5E   1C AE   9E  002F   129          MOVAB   7*4(SP),SP              ;CLEAN STACK
      50   00038278 8F   D0  0033   130          MOVL    #CLI$_SYMABR,R0         ;SET NO SYMBOL ABBREVIATIONS STATUS
                         05  003A   131          RSB
                             003B   132
                             003B   133  ;
                             003B   134  ;        (SP) = # NONUNIQUE CHARACTERS (AFTER *)
                             003B   135  ;        4(SP) = TYPE OF SYMBOL VALUE
                             003B   136  ;        8(SP) = DESCRIPTOR OF SYMBOL VALUE
                             003B   137  ;        16(SP) = DESCRIPTOR OF SYMBOL NAME
                             003B   138  ;        24(SP) = ADDRESS OF SYMBOL TABLE LISTHEAD
                             003B   139  ;
                             003B   140  ALLOCSYM:
                     55  D4  003B   141          CLRL    R5                      ;START WITH FULL SYMBOL NAME
            51   10 AE   7D  003D   142  10$:    MOVQ    16(SP),R1               ;RESET SYMBOL PARAMETERS
               51    55  C2  0041   143          SUBL    R5,R1                   ;FIND SIZE TO SEARCH FOR THIS TIME
            50   18 AE   D0  0044   144          MOVL    24(SP),R0               ;SET ADDRESS OF SYMBOL TABLE LISTHEAD
                 013E    30  0048   145          BSBW    DCL$SEARCHT             ;SEARCH FOR SYMBOL
                 11  50  E9  004B   146          BLBC    R0,30$                  ;IF LBC SEARCH FAILURE
            01   0A A3   91  004E   147          CMPB    SYM_B_TYPE(R3),#SYM_K_PERM ;PERMANENT SYMBOL?
                     74  13  0052   148          BEQL    SYMOVF                  ;IF YES - REFUSE TO ALLOCATE
                             0054   149          ASSUME  SYM_B_NONUNIQUE EQ SYM_T_SYMBOL-1
         51    74    51  83  0054   150          SUBB3   R1,-(R4),R1             ;FIND SYMBOL'S NEW
                     51  97  0058   151          DECB    R1                      ; UNIQUENESS POINT
                     63  1E  005A   152          BGEQU   ABSYMD                  ;IF GEQ, AMBIGUOUS SYMBOL
                   0086  30  005C   153          BSBW    DCL$DEALLOCSYM          ;DEALLOCATE SYMBOL ENTRY
            DA 55    6E  F3  005F   154  30$:    AOBLEQ  (SP),R5,10$             ;LOOP IF MORE SYMBOLS TO CHECK
                             0063   155
   51    10 AE   08 AE   C1  0063   156          ADDL3   8(SP),16(SP),R1         ;CALCULATE LENGTH OF SYMBOL STRINGS
         02      04 AE   D1  0069   157          CMPL    4(SP),#SYM_K_BINARY     ;BINARY VALUE?
                     05  12  006D   158          BNEQ    40$                     ;BRANCH IF NOT
      51    10 AE    04  C1  006F   159          ADDL3   #4,16(SP),R1            ;SET LENGTH OF SYMBOL PLUS LONGWORD
               51    0F  C0  0074   160  40$:    ADDL    #SYM_T_SYMBOL+3,R1      ;ADD IN FIXED OVERHEAD AND ROUND
                     5C  10  0077   161          BSBB    DCL$ALCDYNMEM           ;ALLOCATE DYNAMIC MEMORY
                 4C  50  E9  0079   162          BLBC    R0,SYMOVF               ;IF LBC ALLOCATION FAILURE
            08 A2    51  D0  007C   163          MOVL    R1,SYM_W_SIZE(R2)       ;SET SIZE OF ALLOCATED BLOCK, ETC.
            0B A2    8E  F6  0080   164          CVTLB   (SP)+,SYM_B_NONUNIQUE(R2) ;SET UNIQUENESS POINT
            0A A2    8E  F6  0084   165          CVTLB   (SP)+,SYM_B_TYPE(R2)    ;SET SYMBOL VALUE TYPE
            10 BE    62  0E  0088   166          INSQUE  SYM_L_FL(R2),@16(SP)    ;INSERT ENTRY IN SYMBOL TABLE
            10 AE    52  D0  008C   167          MOVL    R2,16(SP)               ;SAVE ADDRESS OF NEW ENTRY
            53   08 AE   7D  0090   168          MOVQ    8(SP),R3                ;GET SYMBOL NAME
            0C A2    53  90  0094   169          MOVB    R3,SYM_T_SYMBOL(R2)     ;INSERT LENGTH OF SYMBOL
      0D A2    64    53  28  0098   170          MOVC    R3,(R4),SYM_T_SYMBOL+1(R2) ;INSERT SYMBOL NAME
               51    6E  7D  009D   171          MOVQ    (SP),R1                 ;GET SYMBOL VALUE
            50   10 AE   D0  00A0   172          MOVL    16(SP),R0               ;RETRIEVE ADDRESS OF ENTRY
            02   0A A0   91  00A4   173          CMPB    SYM_B_TYPE(R0),#SYM_K_BINARY ;BINARY VALUE?
                     05  12  00A8   174          BNEQ    50$                     ;BRANCH IF STRING VALUE
               83    51  D0  00AA   175          MOVL    R1,(R3)+                ;STORE LONGWORD BINARY VALUE
                     07  11  00AD   176          BRB     60$
               83    51  B0  00AF   177  50$:    MOVW    R1,(R3)+                ;INSERT LENGTH OF STRING VALUE
         63    62    51  28  00B2   178          MOVC    R1,(R2),(R3)            ;INSERT STRING VALUE
                     3E  BA  00B6   179  60$:    POPR    #^M<R1,R2,R3,R4,R5>     ;RESTORE REGISTERS
               51    55  D0  00B8   180          MOVL    R5,R1                   ;RETURN ADDRESS OF SYMBOL ENTRY
               50    01  D0  00BB   181          MOVL    #1,R0                   ;SET SUCCESS INDICATOR
                         05  00BE   182          RSB                             ;
                             00BF   183
                             00BF   184          .ENABL  LSB
                             00BF   185
```

```
                        00BF   186 ABSYMD: STATUS  ABSYMD                  ;SET AMBIGUOUS SYMBOL STATUS
             07    11   00C6   187         BRB     90$
                        00C8   188
                        00C8   189 SYMOVF: STATUS  SYMOVF                  ;SET SYMBOL TABLE OVERFLOW STATUS
        5E   08    C0   00CF   190 90$:    ADDL    #8,SP                   ;POP OFF TOP 2 LONGWORDS
             3E    BA   00D2   191         POPR    #^M<R1,R2,R3,R4,R5>     ;RESTORE REGISTERS
                   05   00D4   192         RSB
                        00D5   193
                        00D5   194         .DSABL  LSB
                        00D5     ;+
                        00D5     ; DCL$ALLDYNMEM - DISPATCH TO MEMORY ALLOCATION SUBROUTINE
                        00D5     ;
                        00D5     ; ENTER HERE TO DISPATCH TO THE MEMORY ALLOCATION ROUTINE
                        00D5     ;
                        00D5     ; INPUT:
                        00D5     ;
                        00D5     ;       R11 = ADDRESS OF PROCESS WORK AREA
                        00D5     ;       R1 = SIZE OF BLOCK
                        00D5     ;
                        00D5     ; OUTPUTS:
                        00D5     ;
                        00D5     ;       R1 = SIZE OF BLOCK ALLOCATED
                        00D5     ;       R2 = ADDRESS OF BLOCK
                        00D5     ;       R3 IS DESTROYED
                        00D5     ;-
                        00D5
                        00D5     DCL$ALLDYNMEM::                                 ;ALLOCATE DYNAMIC MEMORY
      53    20 AB   9E  00D5             MOVAB   PRC_Q_ALLOCREG(R11),R3  ;GET ADDRESS OF ALLOCATION LISTHEAD
         51    07   C0  00D9             ADDL    #7,R1                   ;ROUND UP TO QUADWORD BOUNDARY
         51    07   CA  00DC             BICL    #7,R1                   ;TRUNCATE TO QUADWORD MULTIPLE
      00000000'9F   17  00DF             JMP     @#EXE$ALLOCATE          ;ALLOCATE SYMBOL TABLE ENTRY
```

```
                      00E5          .SBTTL  DEALLOCATE SYMBOL TABLE ENTRY
                      00E5     ;+
                      00E5     ; DCL$DEALLOCSYM - DEALLOCATE SYMBOL TABLE ENTRY
                      00E5     ; DCL$DEADYNMEM  - DEALLOCATE DYNAMIC MEMORY
                      00E5     ;
                      00E5     ; THIS ROUTINE IS CALLED TO REMOVE A SYMBOL FROM ITS TABLE AND/OR
                      00E5     ; DEALLOCATE DYNAMIC MEMORY USED.
                      00E5     ;
                      00E5     ; INPUTS TO DEALLOCSYM:
                      00E5     ;
                      00E5     ;        R11 = ADDRESS OF PROCESS WORK AREA
                      00E5     ;        R3 = ADDRESS OF SYMBOL ENTRY.
                      00E5     ;
                      00E5     ; INPUTS TO DEALDYNMEM:
                      00E5     ;
                      00E5     ;        R11 = ADDRESS OF PROCESS WORK AREA
                      00E5     ;        R0 = ADDRESS OF BLOCK TO DEALLOCATE
                      00E5     ;        R1 = SIZE OF BLOCK
                      00E5     ;
                      00E5     ;        IT IS ASSUMED THAT CONTROL Y/C AST'S ARE DISABLED.
                      00E5     ;
                      00E5     ; OUTPUTS:
                      00E5     ;
                      00E5     ;        THE SYMBOL IS REMOVED FROM ITS TABLE AND/OR ITS STORAGE IS DEALLOCATED.
                      00E5     ;-
                      00E5
                      00E5     DCL$DEALLOCSYM::                                  ;DEALLOCATE SYMBOL TABLE ENTRY
      50   63    0F   00E5             REMQUE   SYM_L_FL(R3),R0                  ;REMOVE SYMBOL ENTRY FROM ITS TABLE
  51   08 A0     3C   00E8             MOVZWL   SYM_W_SIZE(R0),R1                ;GET SIZE OF BLOCK TO DEALLOCATE
                      00EC     DCL$DEADYNMEM::                                   ;DEALLOCATE DYNAMIC MEMORY
      51   07    C0   00EC             ADDL     #7,R1                            ;ROUND UP TO QUADWORD BOUNDARY
      51   07    CA   00EF             BICL     #7,R1                            ;TRUNCATE TO QUADWORD MULTIPLE
  53   20 AB     9E   00F2             MOVAB    PRC_Q_ALLOCREG(R11),R3           ;GET ADDRESS OF ALLOCATION LISTHEAD
00000000'9F      17   00F6             JMP      @#EXE$DEALLOCATE                 ;DEALLOCATE SYMBOL ENTRY STORAGE
```

```
                  00FC          .SBTTL  CONVERT SYMBOL VALUE TO STRING
                  00FC  ;+
                  00FC  ; DCL$SYM_STRING - GET SYMBOL VALUE AND CONVERT TO A STRING
                  00FC  ;
                  00FC  ; THIS ROUTINE IS CALLED TO SEARCH ALL LOCAL AND GLOBAL SYMBOL TABLES FOR
                  00FC  ; A SPECIFIED SYMBOL, AND TO RETURN THE STRING FORM OF THE SYMBOL VALUE.
                  00FC  ; THAT IS, IF THE SYMBOL VALUE IS NUMERIC, IT IS CONVERTED TO ASCII DECIMAL
                  00FC  ; BEFORE BEING RETURNED.
                  00FC  ;
                  00FC  ; INPUTS:
                  00FC  ;
                  00FC  ;       R11 = ADDRESS OF PROCESS WORK AREA
                  00FC  ;
                  00FC  ;       R1 = LENGTH OF SYMBOL.
                  00FC  ;       R2 = ADDRESS OF SYMBOL.
                  00FC  ;
                  00FC  ; OUTPUTS:
                  00FC  ;
                  00FC  ;       R0 = STATUS
                  00FC  ;       R1 = LENGTH OF VALUE STRING
                  00FC  ;       R2 = ADDRESS OF VALUE STRING
                  00FC  ;       R3 = DESTROYED
                  00FC  ;       R4 = TABLE FLAG
                  00FC  ;               1 ==> FOUND IN LOCAL SYMBOL TABLE
                  00FC  ;               2 ==> FOUND IN GLOBAL SYMBOL TABLE
                  00FC  ;
                  00FC  ;       THE STRING MAY ACTUALLY RESIDE IN EITHER THE SYMBOL TABLE OR
                  00FC  ;       IN THE UNUSED PORTION OF THE EXPANSION BUFFER.  IT IS THE CALLER'S
                  00FC  ;       RESPONSIBILITY TO COPY THE STRING BEFORE USING THE EXPANSION BUFFER.
                  00FC  ;-
                  00FC
                  00FC  DCL$SYM_STRING::
        19    10  00FC          BSBB    DCL$SEARCH              ; SEARCH ALL SYMBOL TABLES
     03 50    E8  00FE          BLBS    R0,DCL$CVT_STRING       ; BRANCH IF NOT FOUND
        51    D4  0101          CLRL    R1                      ; RETURN NULL STRING ON ERROR
              05  0103          RSB
```

```
                    0104            .SBTTL  CONVERT EXPRESSION RESULT TO STRING
                    0104    ;+
                    0104    ; DCL$CVT_STRING - CONVERT EXPRESSION RESULT TO A STRING
                    0104    ;
                    0104    ; IF THE SYMBOL VALUE IS NUMERIC, IT IS CONVERTED TO ASCII DECIMAL.
                    0104    ;
                    0104    ; INPUTS:
                    0104    ;
                    0104    ;       R11 = ADDRESS OF PROCESS WORK AREA
                    0104    ;
                    0104    ;       R1/R2 = QUADWORD DESCRIBING VALUE:
                    0104    ;               IF R2 NONZERO, QUADWORD IS A STRING DESCRIPTOR
                    0104    ;               IF R2 ZERO, R1 IS A BINARY LONGWORD VALUE
                    0104    ;
                    0104    ; OUTPUTS:
                    0104    ;
                    0104    ;       R0 = STATUS
                    0104    ;       R1 = LENGTH OF VALUE STRING
                    0104    ;       R2 = ADDRESS OF VALUE STRING
                    0104    ;       R3 = DESTROYED
                    0104    ;       R4 = PRESERVED
                    0104    ;
                    0104    ;       THE STRING MAY ACTUALLY RESIDE IN EITHER THE SYMBOL TABLE OR
                    0104    ;       IN THE UNUSED PORTION OF THE EXPANSION BUFFER.  IT IS THE CALLER'S
                    0104    ;       RESPONSIBILITY TO COPY THE STRING BEFORE USING THE EXPANSION BUFFER.
                    0104    ;-
                    0104
                    0104    DCL$CVT_STRING::
          52    D5  0104            TSTL    R2                              ; NUMERIC VALUE?
          0B    12  0106            BNEQ    90$                             ; BRANCH IF STRING
    50    51    D0  0108            MOVL    R1,R0                           ; GET BINARY VALUE
          54    DD  010B            PUSHL   R4                              ; SAVE R4 (JUST IN CASE)
       FEF0'    30  010D            BSBW    DCL$CBTA_DEC                    ; CONVERT TO ASCII IN EXPANSION BUFFER
    54    8E    D0  0110            MOVL    (SP)+, R4                       ; RESTORE SAVED R4
    50    01    D0  0113    90$:    MOVL    #1,R0                           ; SET SUCCESS
                05  0116            RSB
```

```
                     0117   328         .SBTTL  SEARCH FOR SYMBOL ENTRY
                     0117   329 ;+
                     0117   330 ; DCL$SEARCH - SEARCH FOR SYMBOL ENTRY
                     0117   331 ;
                     0117   332 ; THIS ROUTINE IS CALLED TO SEARCH ALL LOCAL AND GLOBAL SYMBOL TABLES FOR
                     0117   333 ; AN ENTRY.  THE LOCAL SYMBOL TABLES FOR EACH OF THE INDIRECT LEVELS ARE
                     0117   334 ; FIRST SEARCHED.  IF NOTHING FOUND, THE GLOBAL SYMBOL TABLE IS SEARCHED.
                     0117   335 ;
                     0117   336 ; INPUTS:
                     0117   337 ;
                     0117   338 ;       R11 = ADDRESS OF PROCESS WORK AREA
                     0117   339 ;
                     0117   340 ;       R1 = LENGTH OF SYMBOL.
                     0117   341 ;       R2 = ADDRESS OF SYMBOL.
                     0117   342 ;
                     0117   343 ; OUTPUTS:
                     0117   344 ;
                     0117   345 ;       R0 = STATUS
                     0117   346 ;       R1/R2 = QUADWORD DESCRIBING SYMBOL VALUE:
                     0117   347 ;               IF R2 NONZERO, QUADWORD IS A STRING DESCRIPTOR
                     0117   348 ;               IF R2 ZERO, R1 IS A BINARY LONGWORD VALUE
                     0117   349 ;       R3 = ADDRESS OF SYMBOL ENTRY.
                     0117   350 ;       R4 = TABLE FLAG
                     0117   351 ;               1 ==> FOUND IN LOCAL SYMBOL TABLE
                     0117   352 ;               2 ==> FOUND IN GLOBAL SYMBOL TABLE
                     0117   353 ;
                     0117   354 ;-
                     0117   355
                     0117   356 DCL$SEARCH::                                     ;SEARCH FOR SYMBOL ENTRY
     0012      30    0117   357         BSBW    DCL$SEARCH_LOCAL                ;SEARCH LOCAL SYMBOL TABLES
  54   01      D0    011A   358         MOVL    #1,R4                           ;INDICATE LOCAL SYMBOL TABLE MATCH FOUND
    0B 50      E8    011D   359         BLBS    R0,10$                          ;IF LBS MATCH FOUND
     005A      30    0120   360         BSBW    DCL$SEARCH_GLOBAL               ;SEARCH GLOBAL SYMBOL TABLE
  54   02      D0    0123   361         MOVL    #2,R4                           ;INDICATE GLOBAL SYMBOL TABLE MATCH FOUND
    02 50      E8    0126   362         BLBS    R0,10$                          ;IF LBS MATCH FOUND
       51      D4    0129   363         CLRL    R1                              ;RETURN NULL STRING IF NO MATCH
               05    012B   364 10$:    RSB                                     ;
```

```
                              012C          .SBTTL  SEARCH FOR SYMBOL ENTRY IN A LOCAL SYMBOL TABLE
                              012C     ;+
                              012C     ; DCL$SEARCH_LOCAL - SEARCH FOR SYMBOL ENTRY IN A LOCAL SYMBOL TABLE
                              012C     ;
                              012C     ; THIS ROUTINE IS CALLED TO SEARCH ALL LOCAL SYMBOL TABLES FOR AN ENTRY.
                              012C     ; THE LOCAL SYMBOL TABLES FOR EACH OF THE INDIRECT LEVELS ARE SEARCHED.
                              012C     ;
                              012C     ; INPUTS:
                              012C     ;
                              012C     ;       R11 = ADDRESS OF PROCESS WORK AREA
                              012C     ;
                              012C     ;       R1 = LENGTH OF SYMBOL.
                              012C     ;       R2 = ADDRESS OF SYMBOL.
                              012C     ;
                              012C     ; OUTPUTS:
                              012C     ;
                              012C     ;       R0 = STATUS
                              012C     ;       R1/R2 = QUADWORD DESCRIBING SYMBOL VALUE:
                              012C     ;               IF R2 NONZERO, QUADWORD IS A STRING DESCRIPTOR
                              012C     ;               IF R2 ZERO, R1 IS A BINARY LONGWORD VALUE
                              012C     ;       R3 = ADDRESS OF SYMBOL ENTRY.
                              012C     ;
                              012C     ;-
                              012C
                              012C     DCL$SEARCH_LOCAL::                               ;SEARCH FOR SYMBOL ENTRY IN LOCAL TABLE
                              012C             DISABLE                                  ;DISABLE CTRL/Y'S TO GUARANTEE INTEGRITY
                              0132                                                      ;OF PRC_Q_LOCAL(R11)
      7E    38 AB     7D      0132             MOVQ    PRC_Q_LOCAL(R11),-(SP)   ;SAVE CURRENT LOCAL SYMBOL TABLE LISTHEAD
          00A0 CB     DD      0136             PUSHL   PRC_L_STACKPT(R11)       ;SAVE ADDRESS OF INDIRECT STACK POINTER
      50    38 AB     9E      013A     5$:     MOVAB   PRC_Q_LOCAL(R11),R0      ;SET ADDRESS OF LOCAL SYMBOL TABLE LISTHEAD
               49     10      013E             BSBB    DCL$SEARCHT              ;SEARCH LOCAL SYMBOL TABLE FOR ENTRY
            15 50     E8      0140             BLBS    R0,10$                   ;IF LBS MATCH FOUND
      50       8E     D0      0143             MOVL    (SP)+,R0                 ;RETRIEVE ADDRESS OF INDIRECT FRAME
            74 A0     9F      0146             PUSHAB  IDF_K_LENGTH(R0)         ;CALCULATE ADDRESS OF NEXT FRAME
38 AB    0084 C0      7D      0149             MOVQ    IDF_Q_LOCAL+IDF_K_LENGTH(R0),PRC_Q_LOCAL(R11) ;REPLACE LISTHEAD
E6 5E A0       00     E0      014F             BBS     #IDF_V_INPOPN,IDF_W_FLAG(R0),5$ ;IF SET, SEARCH NEXT TABLE
               50     D4      0154             CLRL    R0                       ;NO MATCH FOUND
               02     11      0156             BRB     20$
               09     10      0158     10$:    BSBB    GET_VALUE                ;GET THE SYMBOL VALUE
               8E     D5      015A     20$:    TSTL    (SP)+                    ;CLEAN STACK
   38 AB       8E     7D      015C             MOVQ    (SP)+,PRC_Q_LOCAL(R11)   ;RESTORE LOCAL SYMBOL TABLE LISTHEAD
                              0160             ENABLE                           ;REENABLE CTRL/Y'S
                      05      0162             RSB                              ;
                              0163
                              0163     ;
                              0163     ; GET DESCRIPTOR (R1/R2) OF SYMBOL VALUE.
                              0163     ;       R2=0 ==> VALUE IS AN INTEGER
                              0163     ;
                              0163
                              0163     GET_VALUE:
      52    0C A3     9A      0163             MOVZBL  SYM_T_SYMBOL(R3),R2      ;GET LENGTH OF SYMBOL
   52     0D A342     9E      0167             MOVAB   SYM_T_SYMBOL+1(R3)[R2],R2 ;GET ADDRESS OF VALUE LENGTH
   02       0A A3     91      016C             CMPB    SYM_B_TYPE(R3),#SYM_K_BINARY ;NUMERIC BINARY VALUE?
               07     12      0170             BNEQ    10$                      ;BRANCH IF NOT
      51       62     D0      0172             MOVL    (R2),R1                  ;GET LONGWORD BINARY VALUE
               52     D4      0175             CLRL    R2                       ;MARK NOT A STRING
               03     11      0177             BRB     20$
```

```
                 51   82   3C  0179    423 10$:     MOVZWL  (R2)+,R1                ;GET LENGTH OF VALUE
                           05  017C    424 20$:     RSB
                               017D    425
```

```
                        017D                .SBTTL  SEARCH FOR SYMBOL ENTRY IN GLOBAL SYMBOL TABLE
                        017D        ;+
                        017D        ; DCL$SEARCH_GLOBAL - SEARCH FOR SYMBOL ENTRY IN GLOBAL SYMBOL TABLE
                        017D        ;
                        017D        ; THIS ROUTINE IS CALLED TO SEARCH THE GLOBAL SYMBOL TABLE FOR AN ENTRY.
                        017D        ;
                        017D        ; INPUTS:
                        017D        ;
                        017D        ;       R11 = ADDRESS OF PROCESS WORK AREA
                        017D        ;
                        017D        ;       R1 = LENGTH OF SYMBOL.
                        017D        ;       R2 = ADDRESS OF SYMBOL.
                        017D        ;
                        017D        ; OUTPUTS:
                        017D        ;
                        017D        ;       R0 = STATUS
                        017D        ;       R1/R2 = QUADWORD DESCRIBING SYMBOL VALUE:
                        017D        ;               IF R2 NONZERO, QUADWORD IS A STRING DESCRIPTOR
                        017D        ;               IF R2 ZERO, R1 IS A BINARY LONGWORD VALUE
                        017D        ;       R3 = ADDRESS OF SYMBOL ENTRY.
                        017D        ;
                        017D        ;-
                        017D
                        017D        DCL$SEARCH_GLOBAL::                                 ;SEARCH FOR SYMBOL ENTRY IN GLOBAL TABLE
     50   28 AB    9E   017D                MOVAB   PRC_Q_GLOBAL(R11),R0        ;SET ADDRESS OF GLOBAL SYMBOL TABLE LISTHEAD
             06    10   0181                BSBB    DCL$SEARCHT                 ;SEARCH GLOBAL SYMBOL TABLE FOR ENTRY
          02 50    E9   0183                BLBC    R0,10$                      ;IF LBC NO MATCH FOUND
             DB    10   0186                BSBB    GET_VALUE                   ;GET THE SYMBOL VALUE
                   05   0188        10$:    RSB                                 ;
                        0189
```

```
                       0189          .SBTTL  SEARCH SPECIFIC SYMBOL TABLE FOR ENTRY
                       0189    ;+
                       0189    ; DCL$SEARCHT - SEARCH SPECIFIC SYMBOL TABLE FOR ENTRY
                       0189    ;
                       0189    ; THIS ROUTINE IS CALLED TO SEARCH A SPECIFIC SYMBOL TABLE FOR AN ENTRY.
                       0189    ;
                       0189    ; INPUTS:
                       0189    ;
                       0189    ;       R0 = ADDRESS OF SYMBOL TABLE LISTHEAD.
                       0189    ;       R1 = LENGTH OF SYMBOL NAME.
                       0189    ;       R2 = ADDRESS OF SYMBOL NAME.
                       0189    ;
                       0189    ; OUTPUTS:
                       0189    ;
                       0189    ;       R0 LOW BIT CLEAR INDICATES SEARCH FAILURE.
                       0189    ;
                       0189    ;               R1 = LENGTH OF SYMBOL NAME.
                       0189    ;               R2 = ADDRESS OF SYMBOL NAME.
                       0189    ;               R3,R4 ARE DESTROYED.
                       0189    ;
                       0189    ;       R0 LOW BIT SET INDICATES SYMBOL FOUND WITH:
                       0189    ;
                       0189    ;               R1 = LENGTH OF SYMBOL NAME.
                       0189    ;               R2 = ADDRESS OF SYMBOL NAME.
                       0189    ;               R3 = ADDRESS OF SYMBOL ENTRY.
                       0189    ;               R4 = ADDRESS OF SYMBOL NAME STRING (JUST PAST THE COUNT).
                       0189    ;-
                       0189
                       0189    DCL$SEARCHT::                                   ;SEARCH SPECIFIC SYMBOL TABLE FOR ENTRY
      53   40 AB   9E  0189            MOVAB   PRC_Q_KEYPAD(R11),R3    ;GET ADDRESS OF KEYPAD TABLE
         53   50   D1  018D            CMPL    R0,R3                   ;IS IT THE ONE WE WANT?
              2B   13  0190            BEQL    30$                     ;YES, THEN BRANCH
         53   50   D0  0192            MOVL    R0,R3                   ;COPY ADDRESS OF SYMBOL TABLE LISTHEAD
         53   63   D0  0195    10$:    MOVL    SYM_L_FL(R3),R3         ;GET ADDRESS OF NEXT ENTRY
         53   50   D1  0198            CMPL    R0,R3                   ;END OF TABLE?
              1F   13  019B            BEQL    20$                     ;IF EQL YES
      54   0C A3   9E  019D            MOVAB   SYM_T_SYMBOL(R3),R4     ;GET ADDRESS OF SYMBOL NAME
         64   51   91  01A1            CMPB    R1,(R4)                 ;SYMBOL LENGTH TOO LONG?
              EF   1A  01A4            BGTRU   10$                     ;IF GTRU YES
 7E   84   0B A3   83  01A6            SUBB3   SYM_B_NONUNIQUE(R3),(R4)+,-(SP) ;FORM UNIQUE LENGTH
         8E   51   91  01AB            CMPB    R1,(SP)+                ;SYMBOL LENGTH TOO SHORT?
              E5   1F  01AE            BLSSU   10$                     ;IF LSSU YES
              0F   BB  01B0            PUSHR   #^M<R0,R1,R2,R3>        ;SAVE SEARCH PARAMETERS
    64   62   51   29  01B2            CMPC    R1,(R2),(R4)            ;SYMBOLS MATCH?
              0F   BA  01B6            POPR    #^M<R0,R1,R2,R3>        ;RESTORE SEARCH PARAMETERS
              DB   12  01B8            BNEQ    10$                     ;IF NEQ NO
              50   D6  01BA            INCL    R0                      ;SET SUCCESS INDICATOR
                   05  01BC    20$:    RSB                             ;
                       01BD
           FE40'   30  01BD    30$:    BSBW    DCL$FIND_KEYPAD         ;CALL KEYPAD SYMBOL SEARCH ROUTINE
                   05  01C0            RSB                             ;RETURN
```

```
                         01C1             .SBTTL  RESTORE SYMBOL DEFINITION AFTER A SPAWN
                         01C1      ;+
                         01C1      ; DCL$RESTORE_SYM - RESTORE SYMBOL DEFINITION AFTER A SPAWN
                         01C1      ;
                         01C1      ; THIS ROUTINE IS CALLED TO RESTORE A SYMBOL AFTER A SPAWN.
                         01C1      ;
                         01C1      ; INPUTS:
                         01C1      ;
                         01C1      ;       R0 = SYMBOL TYPE
                         01C1      ;       R1/R2 = DESCRIPTOR OF SYMBOL VALUE
                         01C1      ;       R3/R4 = DESCRIPTOR OF SYMBOL NAME
                         01C1      ;       R5 = ADDRESS OF SYMBOL TABLE LISTHEAD
                         01C1      ;       R6 = ADDRESS OF CTX BLOCK
                         01C1      ;       R11 = ADDRESS OF PROCESS WORK AREA
                         01C1      ;
                         01C1      ; OUTPUTS:
                         01C1      ;
                         01C1      ;       THE SYMBOL IS ADDED TO THE TAIL OF THE SYMBOL TABLE.
                         01C1      ;
                         01C1      ;       R0 LOW BIT CLEAR INDICATES ALLOCATION FAILURE WITH CLI$_SYMOVF.
                         01C1      ;       R0 LOW BIT SET INDICATES SUCCESSFUL COMPLETION.
                         01C1      ;-
                         01C1
                         01C1      DCL$RESTORE_SYM::
              1F    BB   01C1             PUSHR   #^M<R0,R1,R2,R3,R4>             ;SAVE THE REGISTERS
                         01C3
                         01C3      ;
                         01C3      ; ALLOCATE THE SYMBOL.
                         01C3      ;
         51   0F    C0   01C3             ADDL    #SYM_T_SYMBOL+3,R1              ;GET SIZE OF SYMBOL NEEDED
         50   02    91   01C6             CMPB    #SYM_K_BINARY,R0                ;BINARY SYMBOL?
              03    12   01C9             BNEQ    5$                              ;NO, THEN SKIP
         51   13    D0   01CB             MOVL    #SYM_T_SYMBOL+3+4,R1            ;SET SIZE OF SYMBOL NEEDED
         51   53    C0   01CE      5$:    ADDL    R3,R1                           ;
            FF01    30   01D1             BSBW    DCL$ALLDYNMEM                   ;ALLOCATE DYNAMIC MEMORY
           43 50    E9   01D4             BLBC    R0,90$                          ;IF LBC ALLOCATION FAILURE
                         01D7
                         01D7      ;
                         01D7      ; INITIALIZE THE STATICALLY PLACED FIELDS AND INSERT IT IN THE LINKED LIST.
                         01D7      ;
      08 A2   51    B0   01D7             MOVW    R1,SYM_W_SIZE(R2)               ;SET SIZE OF ALLOCATED BLOCK
   0B A2   06 A6    90   01DB             MOVB    CTX_B_NONUNIQUE(R6),SYM_B_FLAGS(R2) ;SET KEYPAD FLAGS
      0A A2   6E    90   01E0             MOVB    (SP),SYM_B_TYPE(R2)             ;SET VALUE TYPE
         6E   04    91   01E4             CMPB    #SYM_K_KEYPAD,(SP)              ;KEYPAD SYMBOL?
              06    12   01E7             BNEQ    10$                             ;NO, SKIP
      04 B5   62    0E   01E9             INSQUE  (R2),@4(R5)                     ;INSERT ENTRY AT TAIL OF TABLE
              03    11   01ED             BRB     20$                             ;SKIP
         65   62    0E   01EF      10$:   INSQUE  (R2),(R5)                       ;INSERT ENTRY AT HEAD OF TABLE
                         01F2
                         01F2      ;
                         01F2      ; INITIALIZE THE DYNAMICALLY PLACED ASCIC FIELDS.
                         01F2      ;
      53   0C AE    7D   01F2      20$:   MOVQ    12(SP),R3                       ;GET SYMBOL NAME
      0C A2   53    90   01F6             MOVB    R3,SYM_T_SYMBOL(R2)             ;INSERT LENGTH OF SYMBOL
 0D A2   64   53    28   01FA             MOVC    R3,(R4),SYM_T_SYMBOL+1(R2)      ;INSERT SYMBOL NAME
         6E   02    91   01FF             CMPB    #SYM_K_BINARY,(SP)              ;BINARY VALUE
              06    12   0202             BNEQ    30$                             ;NO, THEN SKIP
            83    04 AE   D0  0204               MOVL    4(SP),(R3)+                       ;INSERT THE VALUE
                     0A   11  0208               BRB     40$                               ;SKIP
            83    04 AE   B0  020A      30$:     MOVW    4(SP),(R3)+                       ;INSERT LENGTH OF VALUE
   63    08 BE    04 AE   28  020E               MOVC    4(SP),@8(SP),(R3)                 ;INSERT SYMBOL VALUE
                     1F   BA  0214      40$:     POPR    #^M<R0,R1,R2,R3,R4>               ;RESTORE THE REGISTERS
               50    01   D0  0216               MOVL    #1,R0                             ;SET SUCCESS INDICATOR
                          05  0219               RSB                                       ;
                              021A
                              021A      ;
                              021A      ; RETURN SYMBOL TABLE OVERFLOW STATUS.
                              021A      ;
                     1F   BA  021A      90$:     POPR    #^M<R0,R1,R2,R3,R4>               ;RESTORE THE REGISTERS
                              021C               STATUS  SYMOVF                            ;SET SYMBOL TABLE OVERFLOW STATUS
                          05  0223               RSB
                          0224                 .SBTTL  DELETE SYMBOL FROM SYMBOL TABLE
                          0224       ;+
                          0224       ; DCL$DELSYM - DELETE SYMBOL FROM SYMBOL TABLE
                          0224       ;
                          0224       ; THIS ROUTINE IS CALLED AS AN INTERNAL COMMAND TO DELETE EITHER A
                          0224       ; SPECIFIC SYMBOL OR ALL SYMBOLS FROM A SPECIFIED SYMBOL TABLE.
                          0224       ;
                          0224       ; INPUTS:
                          0224       ;
                          0224       ;         R8 = ADDRESS OF SCRATCH BUFFER DESCRIPTOR.
                          0224       ;         R9 = ADDRESS OF SCRATCH STACK.
                          0224       ;         R11 = BASE ADDRESS OF PROCESS WORK AREA.
                          0224       ;
                          0224       ; OUTPUTS:
                          0224       ;
                          0224       ;         THE SPECIFIED SYMBOL,IF IT EXISTS, IS DELETED,
                          0224       ;                 ELSE NO SUCH SYMBOL IS RETURNED;
                          0224       ;         OR THE ENTIRE CONTENTS OF THE SPECIFIED TABLE IS DELETED.
                          0224       ;         PERMANENT SYMBOLS ARE NEVER DELETED.
                          0224       ;-
                          0224
                          0224       DCL$DELSYM::                                      ; DELETE SYMBOL FROM TABLE
       57    38 AB    7E  0224               MOVAQ   PRC_Q_LOCAL(R11),R7       ; ASSUME DELETING A SPECIFIC SYMBOL
                58    D4  0228               CLRL    R8                        ; FROM THE LOC. TABLE WITH /NOLOG
    59    FDD2 CF    9E   022A               MOVAB   LOCTAB,R9
              FDCE'   30  022F       10$:    BSBW    DCL$GETDVAL               ; GET NEXT RESULT DESCRIPTOR
          55    04    91  0232               CMPB    #PTR_K_ENDLINE,R5         ; IS THIS THE END OF THE LINE?
                4B    13  0235               BEQL    30$                       ; BR IF YES
          55    03    91  0237               CMPB    #PTR_K_PARAMETR,R5        ; IS THIS A PARAMETER?
                46    13  023A               BEQL    30$                       ; BR IF YES
          55    00    91  023C               CMPB    #PTR_K_COMDQUAL,R5        ; IS THIS A QUALIFIER?
                EE    12  023F               BNEQ    10$                       ; BR IF NO
              FDBC'   30  0241               BSBW    DCL$GETNVAL               ; GET QUALIFIER NUMBER
     00'8F      51    91  0244               CMPB    R1,#CLI$K_DLSY_ALL        ; /ALL?
                33    13  0248               BEQL    20$                       ; BR IF ALL
     00'8F      51    91  024A               CMPB    R1,#CLI$K_DLSY_LOCA       ; /LOCAL?
                22    13  024E               BEQL    15$                       ; BR IF LOCAL TABLE
     00'8F      51    91  0250               CMPB    R1,#CLI$K_DLSY_GLOB       ; /GLOBAL?
                11    13  0254               BEQL    12$                       ; BR IF GLOBAL TABLE
     00'8F      51    91  0256               CMPB    R1,#CLI$K_DLSY_LOG        ; /LOG?
                D3    12  025A               BNEQ    10$                       ; IF NOT, IGNORE
          58    01    8A  025C               BICB    #1,R8                     ; ASSUME /NOLOG
            CD  53    E8  025F               BLBS    R3,10$                    ; IT IS /NOLOG. FLAG OK AS IS
          58    01    88  0262               BISB    #1,R8                     ; IT IS /LOG. SET FLAG ACCORDINGLY
                C8    11  0265               BRB     10$                       ; BR FOR MORE
       57    28 AB    7E  0267       12$:    MOVAQ   PRC_Q_GLOBAL(R11),R7      ; SET PROPER TABLE ADDRESS
    59    FD97 CF    9E   026B               MOVAB   GBLTAB,R9                 ; SET GLOBAL STRING ADDR. FOR /LOG MESS.
                BD    11  0270               BRB     10$                       ; BR FOR MORE
       57    38 AB    7E  0272       15$:    MOVAQ   PRC_Q_LOCAL(R11),R7       ; MAKE SURE TABLE ADDRESS IS CORRECT
    59    FD86 CF    9E   0276               MOVAB   LOCTAB,R9                 ; SET LOCAL STRING ADDR. FOR /LOG MESS.
                82    11  0278               BRB     10$
          58    02    88  027D       20$:    BISB    #2,R8                     ; SET DOING ALL FLAG
                AD    11  0280               BRB     10$
       15 58    01    E0  0282       30$:    BBS     #1,R8,50$                 ; BR IF DOING ALL
          50    57    D0  0286               MOVL    R7,R0                     ; SET TABLE ADDRESS
              FEFD    30  0289               BSBW    DCL$SEARCHT               ; SEARCH THE TABLE FOR THE SYMBOL
          04 50       E9  028C               BLBC    R0,40$                    ; DEALLOCTE
```

```
               15  10  028F              BSBB     55$                            ; TEST AND DEALLOCATE
               33  11  0291              BRB      70$                            ;
                       0293      40$:    STATUS   UNDSYM                         ; NO SUCH SYMBOL
                   05  029A              RSB                                     ; EXIT WITH ERROR STATUS
           53      67  D0  029B  50$:    MOVL     (R7),R3                        ; GET ADDRESS OF NEXT SYMBOL
           53      57  D1  029E          CMPL     R7,R3                          ; DONE?
               23  13  02A1              BEQL     70$                            ; BR IF YES
            F5 AF  9F  02A3              PUSHAB   50$                            ; RETURN ADDRESS FOR DEALLOCATE
     01     0A A3  91  02A6      55$:    CMPB     SYM_B_TYPE(R3),#SYM_K_PERM     ; PERMANENT SYMBOL?
               18  13  02AA              BEQL     60$                            ; BR IF YES - DON'T DEALLOCATE IT
            12 58  E9  02AC              BLBC     R8,59$                         ; SKIP IF /NOLOG
            0C A3  9F  02AF              PUSHAB   SYM_T_SYMBOL(R3)               ; GET SYMBOL NAME
               59  DD  02B2              PUSHL    R9                             ; SET GLOBAL/LOCAL ASCIC ADDR.
         51    02  D0  02B4              MOVL     #2,R1                          ; SET FAO COUNT TO 2
 50    0003DF23 8F  D0  02B7              MOVL     #CLI$_DELSYM,R0                ; SET MESSAGE STATUS
            FD3F'  30  02BE              BSBW     DCL$FORMMSG                    ; OUTPUT THE MESSAGE
             FE21  31  02C1      59$:    BRW      DCL$DEALLOCSYM                 ; DEALLOCATE IT
               8E  D5  02C4      60$:    TSTL     (SP)+                          ; CLEAR INTERMEDIATE RETURN
                       02C6      70$:    STATUS   NORMAL                         ;
                   05  02CD              RSB
                       02CE
                       02CE              .END
```

```
ABSYMD                000000BF R      02    DCL$SEARCH_LOCAL      0000012C RG     02
ALLOCSYM              0000003B R      02    DCL$SYM_STRING        000000FC RG     02
CLI$K_DLSY_ALL        ********     X  02    EXE$ALLOCATE          ********     X  02
CLI$K_DLSY_GLOB       ********     X  02    EXE$DEALLOCATE        ********     X  02
CLI$K_DLSY_LOCA       ********     X  02    GBLTAB                00000006 R      02
CLI$K_DLSY_LOG        ********     X  02    GET_VALUE             00000163 R      02
CLI$_ABSYMD         = 000381A0              IDF_B_OUTFLAGS        00000038
CLI$_DELSYM         = 0003DE23              IDF_C_LENGTH          00000074
CLI$_NORMAL         = 00030001              IDF_K_LENGTH          00000074
CLI$_SYMABR         = 00038278              IDF_L_FILENAME        00000068
CLI$_SYMOVF         = 00038138              IDF_L_INPRABCTX       0000000C
CLI$_UNDSYM         = 00038140              IDF_L_LNK             00000000
CTX_B_ACMODE          00000004              IDF_L_ONCTLY          00000060
CTX_B_CONTINUE        00000012              IDF_L_ONERROR         00000008
CTX_B_FLAGS           0000000E              IDF_L_OUTRABCTX       00000024
CTX_B_KEYLENGTH       00000002              IDF_L_SEARCHCTX       00000064
CTX_B_NFLAGS          00000005              IDF_Q_LABEL           00000018
CTX_B_NONUNIQUE       00000006              IDF_Q_LOCAL           00000010
CTX_B_PROMPTLEN       0000000F              IDF_T_INPDVI          0000003C
CTX_B_SYMTAB          00000004              IDF_T_OUTDVI          00000028
CTX_B_SYMTYPE         00000005              IDF_V_INPOPN        = 00000000
CTX_B_TFLAGS          00000005              IDF_W_FLAG            0000005E
CTX_B_TRANCNT         00000006              IDF_W_INPDID          00000052
CTX_C_HDRLEN          00000033              IDF_W_INPFID          0000004C
CTX_G_PROMPT          00000013              IDF_W_INPIFI          00000004
CTX_K_HDRLEN          00000033              IDF_W_INPRFA          00000058
CTX_L_OUTOFBAND       0000000A              IDF_W_ONLEVEL         00000006
CTX_L_QUOTA           00000008              IDF_W_OUTIFI          00000020
CTX_Q_PROCPRIV        00000002              IDF_W_OUTISI          00000022
CTX_T_CMDSTR          00000002              LOCTAB                00000000 R      02
CTX_T_KEYSTATE        00000003              PRC_B_CONTINUE        000000F3
CTX_T_LNMNAME         00000007              PRC_B_DEFRADIX        000000AE
CTX_T_LNMTABLE        0000000C              PRC_B_EXMDEPMOD       000000AD
CTX_T_LOGNAM          00000005              PRC_B_EXMDEPWID       000000AC
CTX_T_SYMBOL          00000007              PRC_B_EXONLYL         0000012D
CTX_W_ENTSIZE         00000002              PRC_B_FLAGS2          000000AF
CTX_W_PMPTCTRL        00000010              PRC_B_IMGFLAG         00000078
CTX_W_PROT            00000006              PRC_B_OUTFLAGS        0000012C
CTX_W_TYPE            00000000              PRC_B_PROMPTLEN       000000F0
DCL$ALLDYNMEM         000000D5 RG     02    PRC_C_LENGTH          00000534
DCL$ALLOCSYM          00000025 RG     02    PRC_G_COMMANDS        00000133
DCL$ALLOCSYMABR       0000000F RG     02    PRC_G_PROMPT          000000F4
DCL$CBTA_DEC          ********     X  02    PRC_K_LENGTH          00000534
DCL$CVT_STRING        00000104 RG     02    PRC_L_CURRKEY         00000048
DCL$DEADYNMEM         000000EC RG     02    PRC_L_EXMDEPADR       000000A8
DCL$DEALLOCSYM        000000E5 RG     02    PRC_L_EXTARG          00000094
DCL$DELSYM            00000224 RG     02    PRC_L_EXTBLK          0000008C
DCL$DISABLE           ********     X  02    PRC_L_EXTCOD          0000009C
DCL$FIND_KEYPAD       ********     X  02    PRC_L_EXTHND          00000090
DCL$FORMMSG           ********     X  02    PRC_L_EXTPRM          00000098
DCL$GETDVAL           ********     X  02    PRC_L_IDFLNK          000000BC
DCL$GETNVAL           ********     X  02    PRC_L_IMGACTSTS       00000080
DCL$GT_SYMABR         0000000D RG     02    PRC_L_INDCLOCK        0000007C
DCL$RESTORE_SYM       000001C1 RG     02    PRC_L_INDEPTH         0000005C
DCL$SEARCH            00000117 RG     02    PRC_L_INDFAB          0000001C
DCL$SEARCHT           00000189 RG     02    PRC_L_INDINPRAB       00000014
DCL$SEARCH_GLOBAL     0000017D RG     02    PRC_L_INDOUTRAB       00000018
```

```
PRC_L_INPRAB                     00000008          PTR_L_ENTITY                        00000008
PRC_L_LASTKEY                    0000004C          SYMOVF                              000000C8 R     02
PRC_L_LSTSTATUS                  000000B0          SYM_B_FLAGS                         0000000B
PRC_L_ONCTLY                     000000B8          SYM_B_NONUNIQUE                     0000000B
PRC_L_ONERROR                    0000006C          SYM_B_TYPE                          0000000A
PRC_L_OUTOFBAND                  000000B4          SYM_K_BINARY                      = 00000002
PRC_L_OUTRAB                     0000000C          SYM_K_KEYPAD                      = 00000004
PRC_L_OUTRABCTX                  00000118          SYM_K_PERM                        = 00000001
PRC_L_PPFLIST                    00000070          SYM_L_BL                            00000004
PRC_L_RECALLPTR                  0000012F          SYM_L_FL                            00000000
PRC_L_RESTART                    00000058          SYM_T_SYMBOL                        0000000C
PRC_L_SAVAP                      00000000          SYM_W_SIZE                          00000008
PRC_L_SAVFP                      00000004          WRK_B_CMDOPT                        FFFFFFC3
PRC_L_SEVERITY                   00000050          WRK_B_MAXPARM                       FFFFFFD0
PRC_L_SPWN                       000000C0          WRK_B_MINPARM                       FFFFFFD1
PRC_L_STACKLM                    000000A4          WRK_B_PARMCNT                       FFFFFFCE
PRC_L_STACKPT                    000000A0          WRK_B_PARMSUM                       FFFFFFCF
PRC_L_STATUS                     00000054          WRK_B_RECALLCNT                     FFFFFFC5
PRC_L_STS                        00000084          WRK_B_VALLEV                        FFFFFFC4
PRC_L_STV                        00000088          WRK_B_VERBTYP                       FFFFFFC2
PRC_L_SYMBOL                     00000060          WRK_C_LENGTH                        FFFFF486
PRC_L_TMBX                       00000074          WRK_G_BUFFER                        FFFFF492
PRC_L_TRMLIST                    00000010          WRK_G_INPBUF                        FFFFF896
PRC_Q_ALLOCREG                   00000020          WRK_G_RESULT                        FFFFF9B6
PRC_Q_COMMAND                    000000E0          WRK_K_LENGTH                        FFFFF486
PRC_Q_FLUSHTIME                  000000D0          WRK_L_CHARPTR                       FFFFF48E
PRC_Q_GLOBAL                     00000028          WRK_L_DISALLOW                      FFFFFFE6
PRC_Q_IMAGENAME                  000000D8          WRK_L_ERRORRTN                      FFFFF9AE
PRC_Q_KEYPAD                     00000040          WRK_L_EXPANDPTR                     FFFFF486
PRC_Q_LABEL                      00000030          WRK_L_IMAGE                         FFFFFFE2
PRC_Q_LOCAL                      00000038          WRK_L_MARKPTR                       FFFFF48A
PRC_Q_SAVEPRIV                   000000E8          WRK_L_PAROUT                        FFFFFFD2
PRC_T_OUTDVI                     0000011C          WRK_L_PMPTADDR                      FFFFF9A2
PRC_W_ASTIOSB                    000000C6          WRK_L_PROMPTRTN                     FFFFF9A6
PRC_W_ASTRETN                    000000C8          WRK_L_PROPTR                        FFFFFFC6
PRC_W_ASTSTATUS                  000000C4          WRK_L_QUABLK                        FFFFFFCA
PRC_W_ATTMBX                     0000007A          WRK_L_READRTN                       FFFFF9AA
PRC_W_FLAGS                      00000068          WRK_L_RECALLPTR                     FFFFFFEA
PRC_W_INPCHAN                    00000064          WRK_L_RSLEND                        FFFFFFB6
PRC_W_ONLEVEL                    0000006A          WRK_L_RSLNXT                        FFFFFFBA
PRC_W_OUTIFI                     00000114          WRK_L_SAVAP                         FFFFFFF8
PRC_W_OUTISI                     00000116          WRK_L_SAVFP                         FFFFFFFC
PRC_W_OUTMBXCHN                  000000CA          WRK_L_SAVSP                         FFFFFFF4
PRC_W_OUTMBXREF                  000000CE          WRK_L_SIGNALRTN                     FFFFFFD6
PRC_W_OUTMBXSIZ                  000000CC          WRK_L_SPECRTN                       FFFFF9B2
PRC_W_PMPTCTRL                   000000F1          WRK_L_TAB_VEC                       FFFFFFDE
PRC_W_WAITIOSB                   00000066          WRK_L_VERB                          FFFFFFBE
PTR_B_LEVEL                      00000004          WRK_W_FLAGS                         FFFFFFF0
PTR_B_NUMBER                     00000005          WRK_W_FLAGS2                        FFFFFFF2
PTR_B_PARMCNT                    00000006          WRK_W_IMGCHAN                       FFFFFFEE
PTR_B_VALUE                      00000000          WRK_W_PMPTLEN                       FFFFF99E
PTR_C_LENGTH                     0000000C
PTR_K_COMDQUAL                 = 00000000
PTR_K_ENDLINE                  = 00000004
PTR_K_LENGTH                     0000000C
PTR_K_PARAMETR                 = 00000003
PTR_L_DESCR                      00000000
```

```
                                             +----------------+
                                             ! Psect synopsis !
                                             +----------------+

PSECT name                       Allocation          PSECT No.  Attributes
----------                       ----------          ---------  ----------
.  ABS  .                        00000000  (     0.)  00 (  0.)  NOPIC    USR   CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                            FFFFFFFC  (     0.)  01 (  1.)  NOPIC    USR   CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
DCL$ZCODE                        000002CE  (   718.)  02 (  2.)  NOPIC    USR   CON    REL    LCL NOSHR   EXE   RD   NOWRT NOVEC BYTE

                                           +------------------------+
                                           ! Performance indicators !
                                           +------------------------+

Phase                          Page faults     CPU Time       Elapsed Time
-----                          -----------     --------       ------------
Initialization                           9     00:00:00.05    00:00:01.48
Command processing                      80     00:00:00.70    00:00:05.37
Pass 1                                 239     00:00:08.11    00:00:26.87
Symbol table sort                        0     00:00:00.84    00:00:03.02
Pass 2                                 114     00:00:01.88    00:00:07.74
Symbol table output                     25     00:00:00.17    00:00:01.17
Psect synopsis output                    2     00:00:00.02    00:00:00.02
Cross-reference output                   0     00:00:00.00    00:00:00.00
Assembler run totals                   469     00:00:11.80    00:00:45.68

The working set limit was 1350 pages.
39437 bytes (78 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 574 non-local and 34 local symbols.
660 source lines were read in Pass 1, producing 16 object records in Pass 2.
44 pages of virtual memory were used to define 24 macros.

                                          +--------------------------+
                                          ! Macro library statistics !
                                          +--------------------------+

Macro library name                        Macros defined
------------------                        --------------
_$255$DUA28:[SYSLIB]SYSBLDMLB.MLB;1                    0
_$255$DUA28:[DCL.OBJ]DCL.MLB;1                        11
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                         0
_$255$DUA28:[SYSLIB]STARLET.MLB;2                      4
TOTALS (all libraries)                                15

737 GETS were required to define 15 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:SYMBOL/OBJ=OBJ$:SYMBOL MSRC$:SYMBOL/UPDATE=(ENH$:SYMBOL)+EXECML$/LIB+LIB$:DCL/LIB+SYS$LIBRARY:SYSBLDMLB/LIB
```

0074 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

PREFIX
REQ

STACLINT
LIS

STATUS
LIS

ANALYZE
LIS

DCXMSG
LIS

STASTUB
LIS

DCXDEF
MDL

EXPAND
LIS

SYSOUTPUT
LIS

DCXPRVDEF
MDL

STATEMENT
LIS

DCX

COMPRESS
LIS

TRANSFER
LIS

SYMBOL
LIS

DCXSHR
MAP

SUBS
LIS